TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

Alex Ots 206481IADB

Advertising space reservation system for an advertising company

Scope of work in Distributed Systems project

Supervisor: Andres Käver

Alcalá de Henares 2023

Table of Contents

Introduction

Advertisement companies use phone communication to manage clients and their reservations about advertising spaces. This kind of approach is okay when dealing with small clients who buy a few spaces or when there are not many clients buying simultaneously. But when the client wants to buy lots of different spaces in different locations, phone communication makes it hard to choose the spaces one by one. Also, there is a possibility that different salespersons sell the same advertising spaces at the same time. To save clients and salespersons valuable time and to give the clients a better overview of the advertising spaces, a reservation system is needed.

The purpose of this work is to build a system that is capable of giving the company's clients a comprehensive and detailed understanding of the options offered by the company and simplifying work for the companies.

Analysis

Users

The system will have two kinds of users: clients and managers. Clients are the ones making a reservation for advertising space. Managers are advertising company's side users who can be either salespersons or any level managers who will have the right to handle the reservations.

Client

The clients will have only limited access to the data. They can see the map which includes all the advertising spaces the company has to offer. Some of the spaces are marked as free and others as taken. The client cannot see who has reserved the taken spaces. Also, the time period of how far the client can see can be modified by the company side managers so the client would not have the possibility to estimate the company's sales based on how many spaces are free or not.

Clients can choose a preferred period and city and see the available advertising spaces on the map. They can also choose which type of spaces they would like to see. For example, they might only choose bus shelters. By clicking on an advertising space they can see additional information about it, for example, which side of the advertisement carrier is available. Additional information also includes self-made photos (not the Google streetview ones) of the space.

The advertisement spaces can be added to the reservation one by one like the „Add to cart“ in a web store. There will also be presets for different campaigns which will save time for larger clients. For example, a preset could be „Schools“ which would include all the bus shelters in front of schools.

The client can also add a design in PDF format to the reservation. It is not mandatory because many reservations are made before the designs are ready. In that case the client gets a reminder to add the designs before the campaign starts.

After the client has chosen desired advertisement spaces they can send the request.

Manager

Managers will have a thorough view of the calendar. They can see all the advertising spaces listed on a calendar with the confirmed and unconfirmed reservations. From the manager's side, reservations can be modified, deleted, and accepted.

The map view isn't useful for managers because they are familiar with their own locations. So their main tool would be the calendar view, which gathers more information together. But still, the map representation can be used if needed.

They will also have the ability to give rights to the client users. Everyone can make an account on the site but that won't give them the client rights. The manager has to approve the account first. This is needed because we can not allow anyone to see the availability of the advertising spaces. Everything is kept on a need-to-know basis, so only clients approved by a manager can log in and even then they can only see as much as needed for them.

After the manager sees a new pending request, he can look into it and get in contact with the client to discuss the price. Automatic price calculation can not be mainly used because the prices depend on relations with the client, the salesperson's previous work, and other factors that can not be calculated. Nonetheless, there will be an option to add prices for the advertising spaces. If this option is used, the client can see the prices when choosing the advertising spaces, and no further communication would be needed after the manager approves the request.

Use case

The client creates an account on the website. After getting approval on the account the client can see the selection screen. The client chooses conditions for his advertising campaign by selecting dates, desired carrier types, and city. The map which first showed all the existing carriers, now only shows those that match the client's criteria. The client clicks on a carrier and sees additional information like photos of the advertising spaces, which side of the carrier is available, and which types of advertisements can be put there. The client clicks „Add to reservation“ and continues browsing. After adding all desired advertising spaces the client inspects them separately and chooses designs for them, and then confirms the reservation. After that, a salesperson from the advertising company contacts them and offers a price.

External API’s

Google Maps API

Google Maps API is used to get the map representation of the data. From the different maps APIs, it was chosen because it has easy-to-understand documentation and lots of examples. Another advantage is its developer-friendly pricing system.

Scoro API

Scoro is a work management software that is used by an advertising company. The company also uses it to make invoices for clients. Data about advertising spaces can also be included in the invoice. At first, the reservation system is developed to be working independently. In the end, Scoro API is added to synchronize data in the database with the data from the Scoro platform. The reservation system is kept independent from Scoro to keep the possibility to swap Scoro for another management software or just to work independently if needed.

Retrospective

Back end

Back end architecture is dividend into 3 main layers. Data objects are mapped between each layer for optimization.

DAL

The DAL layer deals with data access, seeding, and database changes. The Unit Of Work pattern is used to manage database changes in an organized way. All repositories use the same Unit Of Work instance (uow) to ensure that database modifications are always accurate, they either fail or succeed altogether.

BLL

The BLL layer should contain actual business logic. At the moment the layer is implemented but it just maps the data from BLL layer DTO to Public DTO. API controllers are currently dealing with business logic, but as it grows, the next step should be moving business logic from controllers to the BLL.

API Controllers

API controllers' purpose is to give out data to the front end. As said, at the moment controllers also deal with some business logic. Identity controllers also deal with JSON web tokens and refresh tokens. This logic will probably stay there as it will not be greatly changed in the future.

Front end

Frontend is implemented in React because I needed to practice it for the internship. The frontend application is mainly services, routes and components.

Services

Services make requests against the backend application. The basic functionality of the services is in abstract service classes to avoid code repetition in many different entity services. Axios response interceptor is also added to the BaseService so all the responses are checked for 401 response. In the case of the 401 response, BaseService automatically tries to refresh the token, and if successful, the actual caller won't even know about it. BaseService also adds a base URL and authorization to the request to keep the higher-level classes as clean and simple as possible. TokenService is a helper class to manage the user.

Routes

Login and register routes are the only ones accessible without a confirmed user. After logging in and receiving the JSON web token, the user gets access to different Reservation, Carrier, and account management routes. The main view is the "reservation" which displays all reservations related to the user. By clicking on one, the "reservation/details" view is displayed. It shows additional details about the reservation and also displays the AdSpaces related to it. AdSpaces are grouped by the Carriers. From there it is also possible to go into Carrier details, where photos of AdSpaces can be seen. Editing and creating reservations are implemented on the same route. When clicking "edit" from the details view, the create view opens with prefilled data, including the AdSpaces.

The next step is adding the map views for the listed AdSpaces. Unlike in the original scope, I think that also leaving the possibility to view and pick AdSpaces from the list could be useful. So the edit, reservation details, and carrier details views will have the option to view data either on a map or a list. The fact that AdSpace data in the lists is correctly viewable and changeable, tells us that all the framework around the future map implementation is there.

The photos are currently held in the Cloudinary cloud. The reference is set directly to the AdSpaces. React makes a call to the Cloudinary API with the reference from the AdSpaces to get the image. This approach might be temporary and the decision will be made after some consultation.

Components

Separate components are header, footer and log out. Possibly if the application grows, there will be more components to keep them small and understandable.

ERD entities explanation

Created ERD has user-related entities which are provided by the framework, so these won't be discussed here in detail.

CarrierType

Advertising companies have many different carriers where the advertising spaces stand. Different types of carriers may include bus shelters, beach changing cabins, post boards, different billboards, and so on. CarrierType entity has an attribute Type. The type could be for example „Double beach changing cabin“.

Carrier

Carrier is a specific advertising space carrier that has CarrierType as a foreign key, and different attributes that describe its location. Different nullable descriptions are needed to locate different types of carriers. For example, beach cabins will not have an address, bus stop name, or direction. Attribute „Number“ is for identificators that are already given to carriers by the advertising company.

AdSpace

Carrier has one or many advertising spaces. For example, bus stop has two big displays, A1-sized stickers can be added inside and full-sized stickers can be added to the backside. These are all different advertising spaces. The only attribute is the side. Side is a string which describes advertising spaces side on a carrier. Advertising spaces type is described by the foregin key. The RefToImage attribute keeps the reference to image that is kept on a disk. In this case, keeping the images on a disk is a better solution than keeping them on a database, because the images are only put there once and left there. Keeping them on a database would simplify handling them, but since it is not frequently needed, a cheaper and faster solution is preferred.

AdSpaceType

Advertising spaces could be in different sizes and forms. They could be stickers of different sizes, displays, beach cabin covers, etc. Advertising space type should describe a type and its parameters if possible. One of the attributes is „Description“, where all the details could be explained. This is used because advertising space types could be so different that there are not many common attributes for them. So instead of describing type by attributes, even the more complicated special solutions can be accurately described in a larger string.

AdSpacePrice

AdSpacePrice entity can be used to record advertising space prices. It is only needed when an advertising company wants to use automatic price calculations instead of discussing the prices with the client itself. If the prices change, the old prices EndDate would be set and new price added. In that way, all the previous prices can be seen.

Preset

Presets consist of user-made and premade presets. They will make the reservation process faster for regular customers who already know what they want.

AdSpaceInPreset

Table to connect the advertising spaces with a preset.

Reservation

Reservation has a name, state, creation date, approval date, and end date. Name is how the client names its campaign. The approval date is added when a manager approves the reservation. The end date is when the last advertisement related to that reservation is removed or the reservation is declined for some reason. The state could be for example approved, declined, pending or approved but without designs.

AdSpaceInReservation

Reservation has one or many advertising spaces and this table keeps track of them. Campa

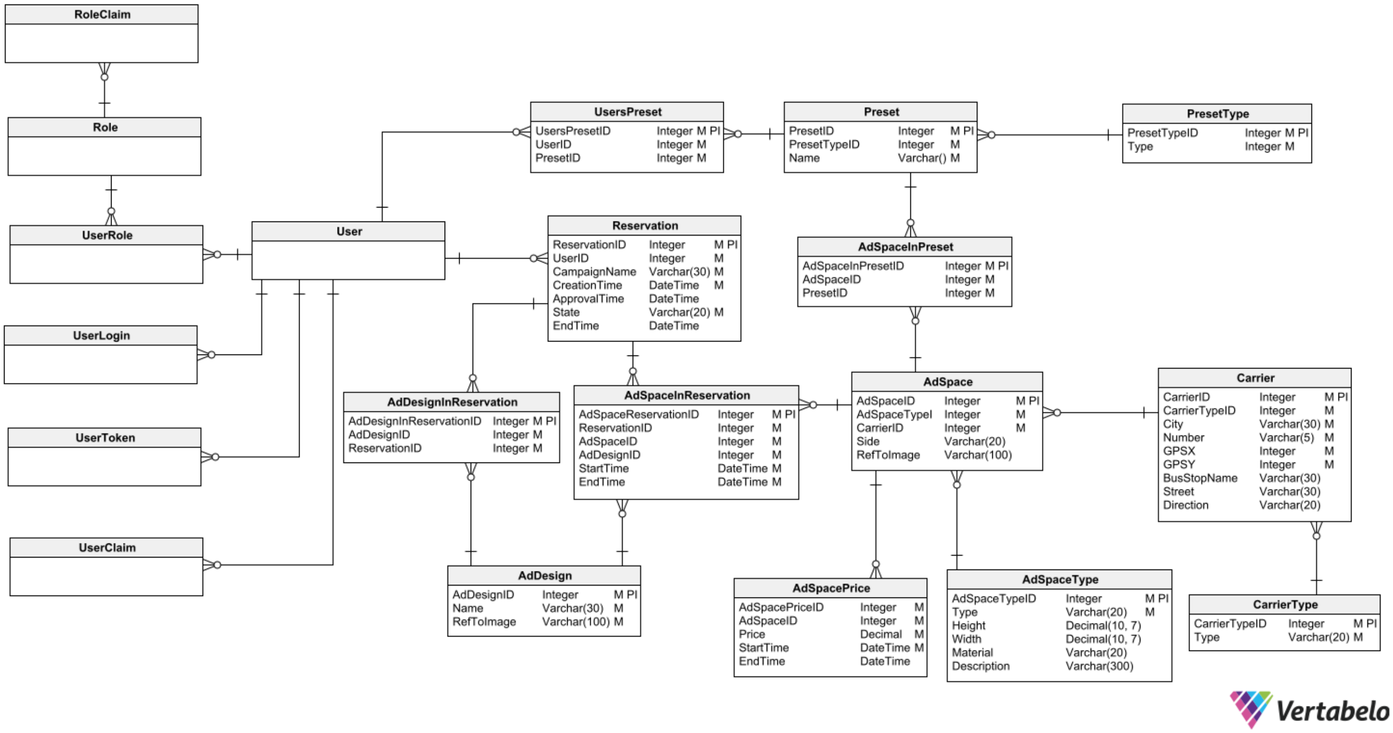
AdDesign

Designs are PDF-files. They will be saved on the disk, and RefToImage attribute will contain their reference.

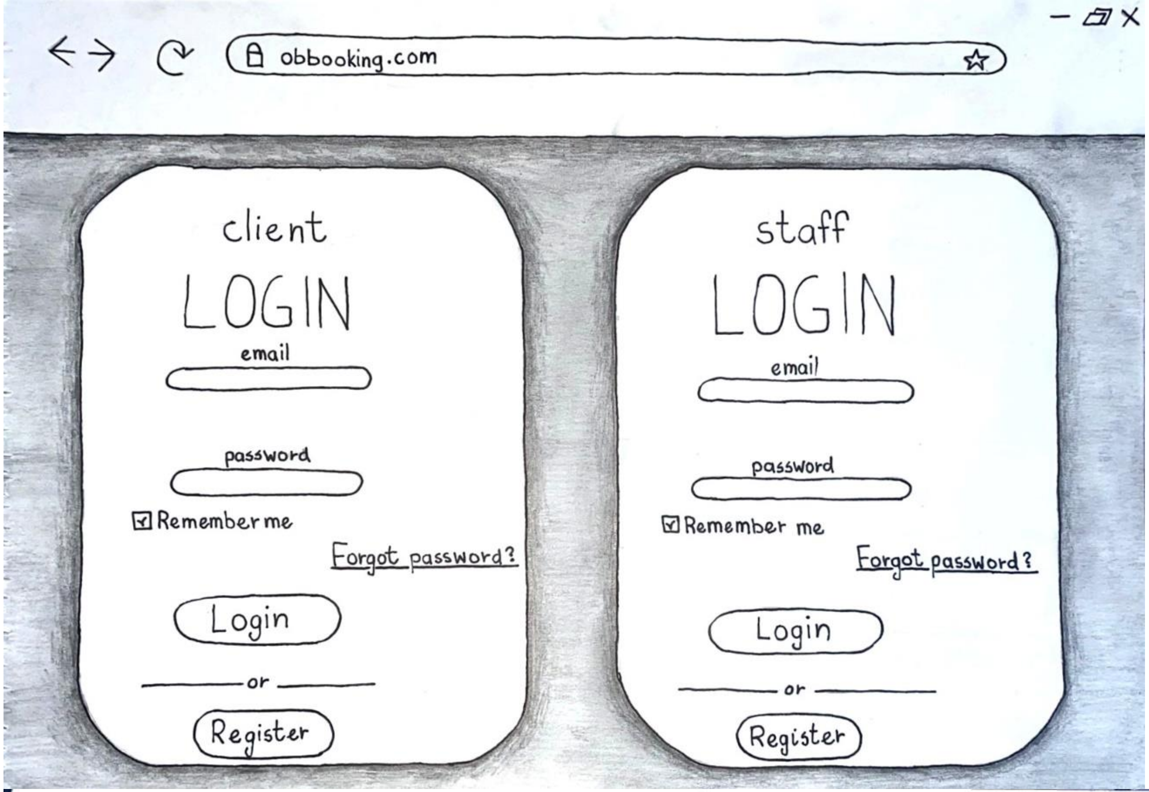
AdDesignInReservation

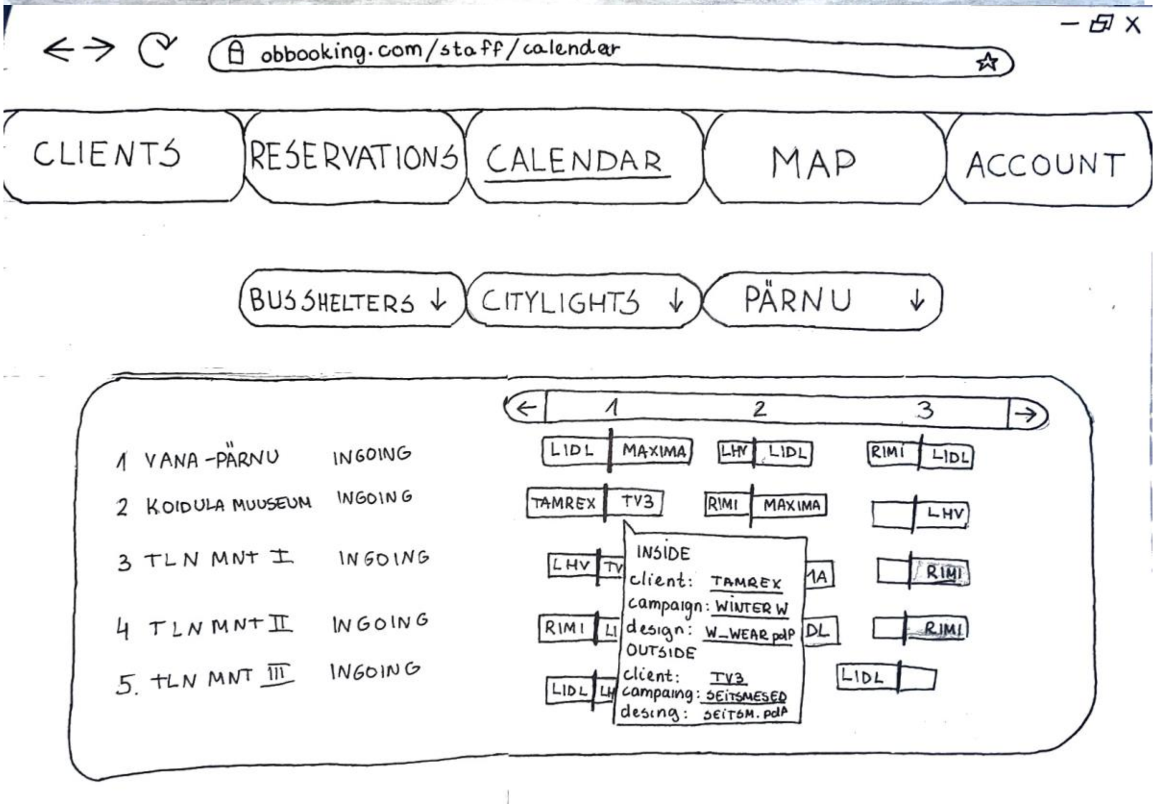
Reservation might have many designs which are distributed to different advertising spaces. The client might also want to use one design in different reservations. AdDesignInReservation holds the data about designs and reservations connections.

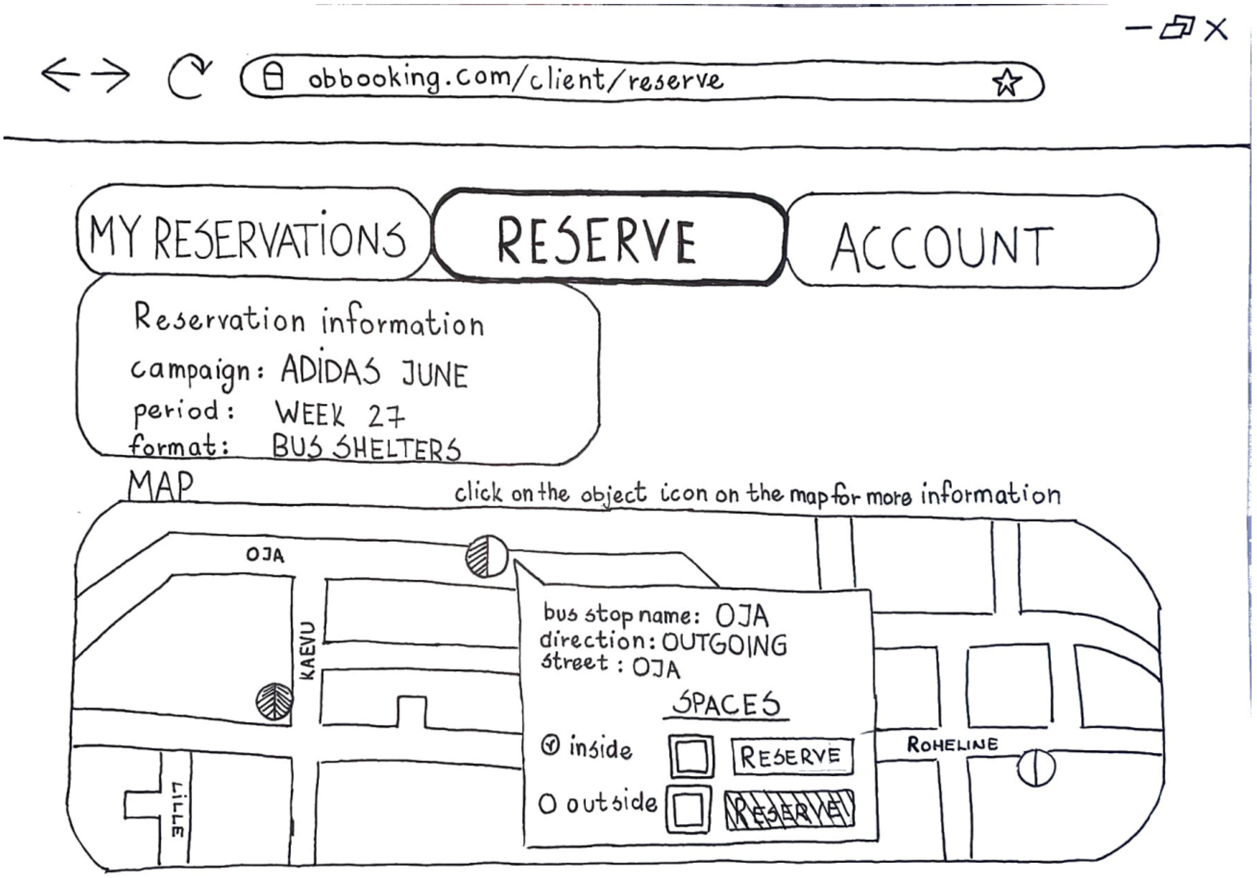
Entity Relationship Diagram



Positive Flow Screens Sketches







Reservation information

campaign: ADIDAS JUNE

period: WEEK 27

format: BUS SHELTERS

MAP

click on the object icon on the map for more information

bus stop name: OJA

direction: OUTGOING

street : OJA

SPACES

☑ inside □ RESERVE

○ outside □ RESERVE